Or Group Design Project Report

Supplemental Documents

Jacob Breiholz Emilio Esteban Gabriel Ritter

jsb4ns@virginia.edu eae4ct@virginia.edu gar5be@virginia.edu

ECE 3663 – Spring 2014  
University of Virginia

This document contains simulations for:

* Worst case delay and maximum frequency of the processor
* Power simulations for the processor and individual components
* Multiplier functionality and critical path

The functionality of the other operations has already been demonstrated in the previous design reviews. They can be viewed again (with their delays) on the following web page:

https://venividiwiki.ee.virginia.edu/mediawiki/index.php/ECE3663S14GroupOR

This document also contains the netlist for the processor. The full netlist with definitions, sizes, buffers, and loads included can also be found on the web page above (otherwise this document would be 35 pages long).

**Processor worst case delay and maximum clock frequency (ADD Operation)**

This simulation was used to determine the total overall delay.

The adder is the slowest block of the ALU, so the critical path goes through it. The worst case inputs for the adder were used, of course.

A was set to 0xFFFF and B goes from 0x0000 to 0x0001.

The output went from 0xFFFF to 0x0000 (Cout went from 0 to 1), as expected.

The delay (minimum clock period) was found to be 0.625ns.

**Complete delay results:**

|  |  |
| --- | --- |
| **Component** | **Delay (ps)** |
| **ADD** | 522 |
| **SUB** | 495 |
| **AND** | 14.4 |
| **OR** | 14.6 |
| **PASS** | 15.1 |
| **SHIFT** | 354 |
| **Multiplier** | 297.9 |
| **Processor** | 625 |

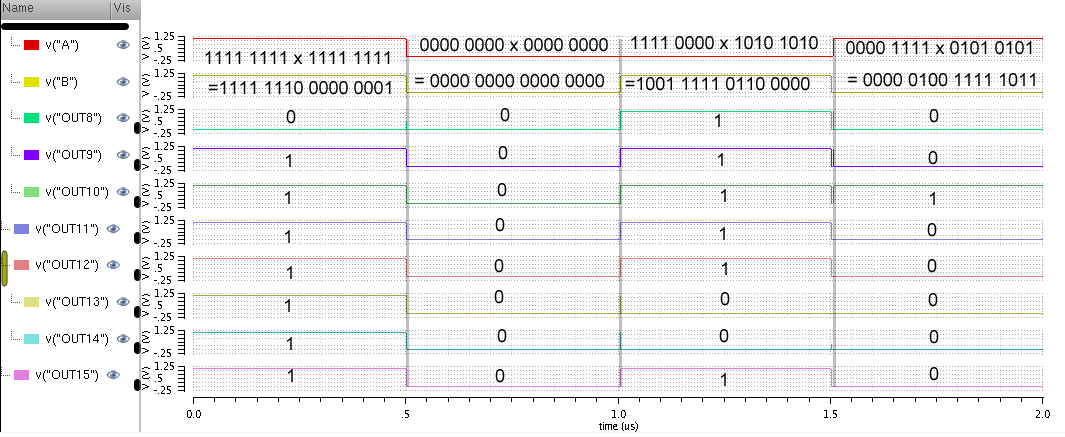
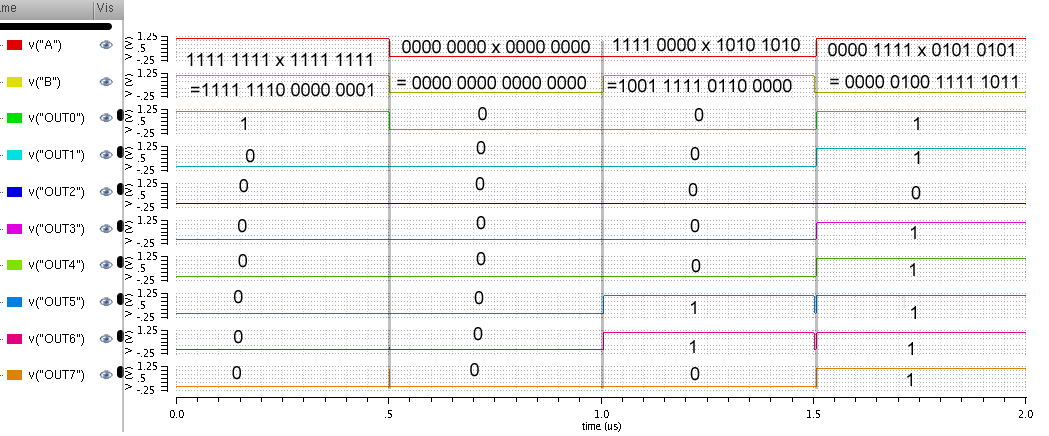
**Processor Energy Simulation**

****

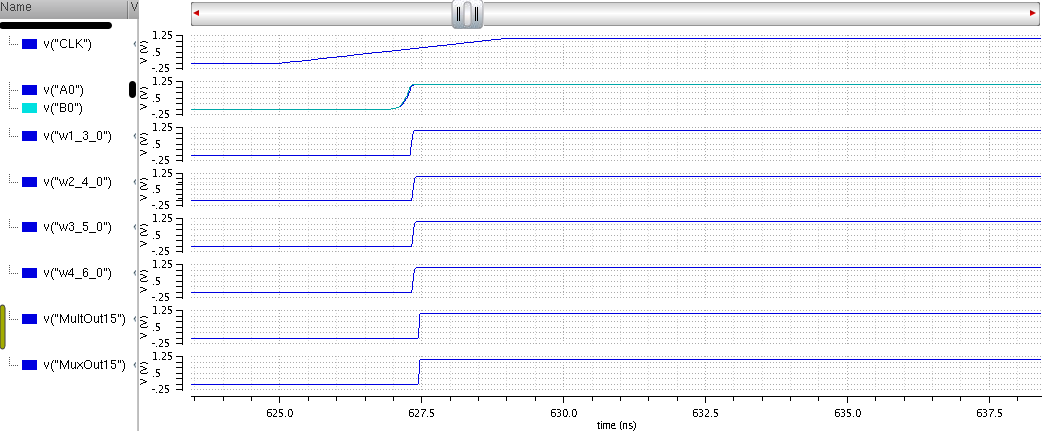
Tests the energy cases specified in the instructions for all of the operations.

**Results:**

|  |  |
| --- | --- |
| **Component** | **Power (W)** |
| **ADD** | 4.904\*10-5 |
| **SUB** | 4.691\*10-5 |
| **AND** | 6.685\*10-6 |
| **OR** | 1.665\*10-5 |
| **PASS** | 2.299\*10-11 |
| **SHIFT** | 3.852\*10-6 |
| **Mux** | 1.572\*10-5 |
| **Register** | 1.115\*10-4 |
| **Multiplier** | 2.450\*10-5 |
| **Processor** | 2.503\*10-4 |

**Multiplier Functionality Simulations**

These two simulations confirm the functionality of the multiplier. The top shows output bits 0 through 7 and the bottom shows bits 8 through 15. Two sources are used to drive the inputs through a total of four different multiplication operations. In all cases, the outputs seen are the outputs expected of the device, and have been noted as such on the diagram.

**Multiplier critical path**

The input to the ALU (and therefore the multiplier) was toggled from all zeroes to all ones. This would have the effect of propagating carries through the Wallace Tree compression stages and through the fast adder. The expected effect would be for all values shown on this graph, namely the inputs, compression stages, and output (ultimately the most significant bit of the multiplier), to transition to 1 with the rising edge of the clock.

**Processor Netlist:**

myAdder (A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 A10 A11 A12 A13 A14 A15 B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 B10 B11 B12 B13 B14 B15 AdderCarryin AdderCout AdderSum0 AdderSum1 AdderSum2 AdderSum3 AdderSum4 AdderSum5 AdderSum6 AdderSum7 AdderSum8 AdderSum9 AdderSum10 AdderSum11 AdderSum12 AdderSum13 AdderSum14 AdderSum15 VDD VSS) Adder\_16b m=1

mySubtractor (A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 A10 A11 A12 A13 A14 A15 B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 B10 B11 B12 B13 B14 B15 SubtractorCout SubtractorSum0 SubtractorSum1 SubtractorSum2 SubtractorSum3 SubtractorSum4 SubtractorSum5 SubtractorSum6 SubtractorSum7 SubtractorSum8 SubtractorSum9 SubtractorSum10 SubtractorSum11 SubtractorSum12 SubtractorSum13 SubtractorSum14 SubtractorSum15 VDD VSS) Subtractor\_16b m=1

myAND (A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 A10 A11 A12 A13 A14 A15 B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 B10 B11 B12 B13 B14 B15 VDD VSS AndOut0 AndOut1 AndOut2 AndOut3 AndOut4 AndOut5 AndOut6 AndOut7 AndOut8 AndOut9 AndOut10 AndOut11 AndOut12 AndOut13 AndOut14 AndOut15) ece3663AND\_16b m=1

myOR (A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 A10 A11 A12 A13 A14 A15 B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 B10 B11 B12 B13 B14 B15 VDD VSS OrOut0 OrOut1 OrOut2 OrOut3 OrOut4 OrOut5 OrOut6 OrOut7 OrOut8 OrOut9 OrOut10 OrOut11 OrOut12 OrOut13 OrOut14 OrOut15) ece3663OR\_16b m=1

myPASSA (A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 A10 A11 A12 A13 A14 A15 VDD VSS PassAOut0 PassAOut1 PassAOut2 PassAOut3 PassAOut4 PassAOut5 PassAOut6 PassAOut7 PassAOut8 PassAOut9 PassAOut10 PassAOut11 PassAOut12 PassAOut13 PassAOut14 PassAOut15) ece3663PASSA\_16b m=1

myShifter (B0 B1 VSS VDD A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 A10 A11 A12 A13 A14 A15 ShifterOut0 ShifterOut1 ShifterOut2 ShifterOut3 ShifterOut4 ShifterOut5 ShifterOut6 ShifterOut7 ShifterOut8 ShifterOut9 ShifterOut10 ShifterOut11 ShifterOut12 ShifterOut13 ShifterOut14 ShifterOut15) shifter m=1

myMult (VDD VSS A0 A1 A2 A3 A4 A5 A6 A7 B0 B1 B2 B3 B4 B5 B6 B7 \

MultCout MultOut0 MultOut1 MultOut2 MultOut3 MultOut4 MultOut5 MultOut6 MultOut7 MultOut8 MultOut9 MultOut10 MultOut11 MultOut12 MultOut13 MultOut14 MultOut15) ece3663multiplier

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Muxs\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

myMux0 (AdderSum0 SubtractorSum0 ShifterOut0 AndOut0 OrOut0 PassAOut0 MultOut0 Out0 VDD VSS MuxOut0 S0 S1 S2) mux m=1

myMux1 (AdderSum1 SubtractorSum1 ShifterOut1 AndOut1 OrOut1 PassAOut1 MultOut1 Out1 VDD VSS MuxOut1 S0 S1 S2) mux m=1

myMux2 (AdderSum2 SubtractorSum2 ShifterOut2 AndOut2 OrOut2 PassAOut2 MultOut2 Out2 VDD VSS MuxOut2 S0 S1 S2) mux m=1

myMux3 (AdderSum3 SubtractorSum3 ShifterOut3 AndOut3 OrOut3 PassAOut3 MultOut3 Out3 VDD VSS MuxOut3 S0 S1 S2) mux m=1

myMux4 (AdderSum4 SubtractorSum4 ShifterOut4 AndOut4 OrOut4 PassAOut4 MultOut4 Out4 VDD VSS MuxOut4 S0 S1 S2) mux m=1

myMux5 (AdderSum5 SubtractorSum5 ShifterOut5 AndOut5 OrOut5 PassAOut5 MultOut5 Out5 VDD VSS MuxOut5 S0 S1 S2) mux m=1

myMux6 (AdderSum6 SubtractorSum6 ShifterOut6 AndOut6 OrOut6 PassAOut6 MultOut6 Out6 VDD VSS MuxOut6 S0 S1 S2) mux m=1

myMux7 (AdderSum7 SubtractorSum7 ShifterOut7 AndOut7 OrOut7 PassAOut7 MultOut7 Out7 VDD VSS MuxOut7 S0 S1 S2) mux m=1

myMux8 (AdderSum8 SubtractorSum8 ShifterOut8 AndOut8 OrOut8 PassAOut8 MultOut8 Out8 VDD VSS MuxOut8 S0 S1 S2) mux m=1

myMux9 (AdderSum9 SubtractorSum9 ShifterOut9 AndOut9 OrOut9 PassAOut9 MultOut9 Out9 VDD VSS MuxOut9 S0 S1 S2) mux m=1

myMux10 (AdderSum10 SubtractorSum10 ShifterOut10 AndOut10 OrOut10 PassAOut10 MultOut10 Out10 VDD VSS MuxOut10 S0 S1 S2) mux m=1

myMux11 (AdderSum11 SubtractorSum11 ShifterOut11 AndOut11 OrOut11 PassAOut11 MultOut11 Out11 VDD VSS MuxOut11 S0 S1 S2) mux m=1

myMux12 (AdderSum12 SubtractorSum12 ShifterOut12 AndOut12 OrOut12 PassAOut12 MultOut12 Out12 VDD VSS MuxOut12 S0 S1 S2) mux m=1

myMux13 (AdderSum13 SubtractorSum13 ShifterOut13 AndOut13 OrOut13 PassAOut13 MultOut13 Out13 VDD VSS MuxOut13 S0 S1 S2) mux m=1

myMux14 (AdderSum14 SubtractorSum14 ShifterOut14 AndOut14 OrOut14 PassAOut14 MultOut14 Out14 VDD VSS MuxOut14 S0 S1 S2) mux m=1

myMux15 (AdderSum15 SubtractorSum15 ShifterOut15 AndOut15 OrOut15 PassAOut15 MultOut15 Out15 VDD VSS MuxOut15 S0 S1 S2) mux m=1

//cout bit passes AdderCout if adder selected, SubtractorCout if subtractor selected, multcout if mult,and VSS if anything else if selected

myMux16 (AdderCout SubtractorCout VSS VSS VSS VSS MultCout VSS VDD VSS MuxCout S0 S1 S2) mux m=1

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Registers\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

myRegisterOut (VDD VSS CLK MuxOut0 MuxOut1 MuxOut2 MuxOut3 MuxOut4 MuxOut5 MuxOut6 MuxOut7 MuxOut8 MuxOut9 MuxOut10 MuxOut11 MuxOut12 MuxOut13 MuxOut14 MuxOut15 Out0 Out1 Out2 Out3 Out4 Out5 Out6 Out7 Out8 Out9 Out10 Out11 Out12 Out13 Out14 Out15) ece3663Register\_16b m=1

myRegisterA (VDD VSS CLK Ain0 Ain1 Ain2 Ain3 Ain4 Ain5 Ain6 Ain7 Ain8 Ain9 Ain10 Ain11 Ain12 Ain13 Ain14 Ain15 A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 A10 A11 A12 A13 A14 A15) ece3663Register\_16b m=1

myRegisterB (VDD VSS CLK Bin0 Bin1 Bin2 Bin3 Bin4 Bin5 Bin6 Bin7 Bin8 Bin9 Bin10 Bin11 Bin12 Bin13 Bin14 Bin15 B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 B10 B11 B12 B13 B14 B15) ece3663Register\_16b m=1